

EXHIBIT C-1

- [首页](#)
- [公司介绍](#)
- [招聘](#)
- [资讯](#)
- [联系我们](#)

- 400-080-9959

公司简介 发展历程 团队介绍 [企业文化](#)

公司简介

深圳积木易搭科技技术有限公司（简称：积木易搭），于2015年创立，总部位于深圳，在武汉和东莞都成立了全资子公司，拥有128位硕士及以上学历的技术研发团队，其中有16位博士和博士后，以及1位博导，科研投入累计超过1亿元，是全球领先的三维数字化解决方案服务商。

积木易搭旗下设有两家全资子公司，分别是 [武汉尺子科技有限公司](#) 与 [深圳爱斯倍科技有限公司](#)。其中，武汉尺子科技负责3D扫描建模底层核心技术研发，包括模型服务平台、核心算法、智能扫描硬件、AI算法研发、数据分析处理等；深圳爱斯倍负责行业应用技术研发，包括3D云设计软件和AR智能营销导购APP的研发等，爱斯倍精研于泛家居领域，并已经推出了服务于家具门店的APP产品“家E搭”。



2015.12

积木易搭在深圳正式成立，进军3D家居设计产业

- 2015.12
- 2016.06
- 2016.12
- 2017.01
- 2017.07
- 2017.12
- 2018.03
- 未来无限可能

团队介绍

孙剑峰

积木易搭创始人、董事长

孙剑峰于1995年创办北京清水室内设计有限公司，有着超过20年家装、地产行业经验和丰富的房地产、家具等产业广泛的资源；2003年收购北京清水爱派建筑设计有限公司，成功于2015年挂牌上市；2010年创办深圳启德投资有限公司，成功投资案例包括：厦门庚能新材料技术有限公司（与中国纳米技术首席科学家危岩联合创办）、武汉巴登城、无锡财富中心等。





姚剑

积木易搭创始人兼CTO & 武汉大学 [计算机视觉与遥感实验室](#) 负责人

姚剑，香港中文大学哲学博士、华中科技大学博士、武汉大学遥感信息工程学院教授、博导及学科带头人，楚天特聘教授；

深圳积木易搭科技技术有限公司创始人兼CTO。

湖北省“楚天学者”特聘教授、中组部第4批“青年千人计划”公示入选者，长沙市引进紧缺急需和战略型人才计划领军人才。

计算机视觉、机器学习、深度学习、人工智能、图像处理、三维重建等领域专家，长期担任相关领域的国际核心期刊和顶级国际学术会议审稿人。

现任中国人工智能学会理事、中国计算机学会会员、中国图象图形学学会会员、CSIG成像探测与感知专委会委员、CCF计算机视觉专业委员会委员。国家重点研发计划智能机器人重点专项课题负责人。

- 产品与服务

- [3D扫描仪](#)
 - [3D模型云平台](#)
 - [行业解决方案](#)
 - [招商加盟](#)

- 关于我们

- [公司简介](#)
 - [招聘](#)
 - [联系我们](#)

- 电话：400-080-9959 邮箱：service@jimuyida.com

地址：深圳市南山区学府路百度国际大厦东塔楼27楼北半层

-

-
-
-

```

void niViewCamera::cameraRotate(const niSceneInfo& scn_info, const niTypes::niPoint2i& pcur, const niTypes::niPoint2i& pprev)
{
    niTypes::niPoint3f rotationPoint;
    scn_info.getCenterPoint(rotationPoint);

    float dx = (float) (pcur.x - pprev.x) * 0.003f;
    float dy = (float) (pcur.y - pprev.y) * 0.003f;

    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glMultMatrixd( getViewMatrixGL() );

    double mat_view[4][4];
    glGetDoublev(GL_MODELVIEW_MATRIX, &mat_view[0][0]);
    glLoadIdentity();
    glTranslatef(rotationPoint.x, rotationPoint.y, rotationPoint.z);
    glRotated( dx*180/_PI, mat_view[0][1], mat_view[1][1], mat_view[2][1]);
    glRotated( dy*180/_PI, mat_view[0][0], mat_view[1][0], mat_view[2][0]);
    glTranslatef(-rotationPoint.x, -rotationPoint.y, -rotationPoint.z);
    double mat_rotate[4][4];
    glGetDoublev(GL_MODELVIEW_MATRIX, &mat_rotate[0][0]);

    glLoadIdentity();
    glLoadMatrixd(&mat_view[0][0]);
    glMultMatrixd(&mat_rotate[0][0]);

    double m[16];
    glGetDoublev(GL_MODELVIEW_MATRIX, m);
    setViewMatrixGL(m);
}

```

Precession occurs here :

```
glRotated( dx*180/_PI, mat_view[0][1], mat_view[1][1], mat_view[2][1]);
```

```
glRotated( dy*180/_PI, mat_view[0][0], mat_view[1][0], mat_view[2][0]);
```

After the first rotation by dx, matrix view will change and become mat_view_dx, but in the code the next dy rotation is counted using to the old mat_view basis that was before the first rotation.